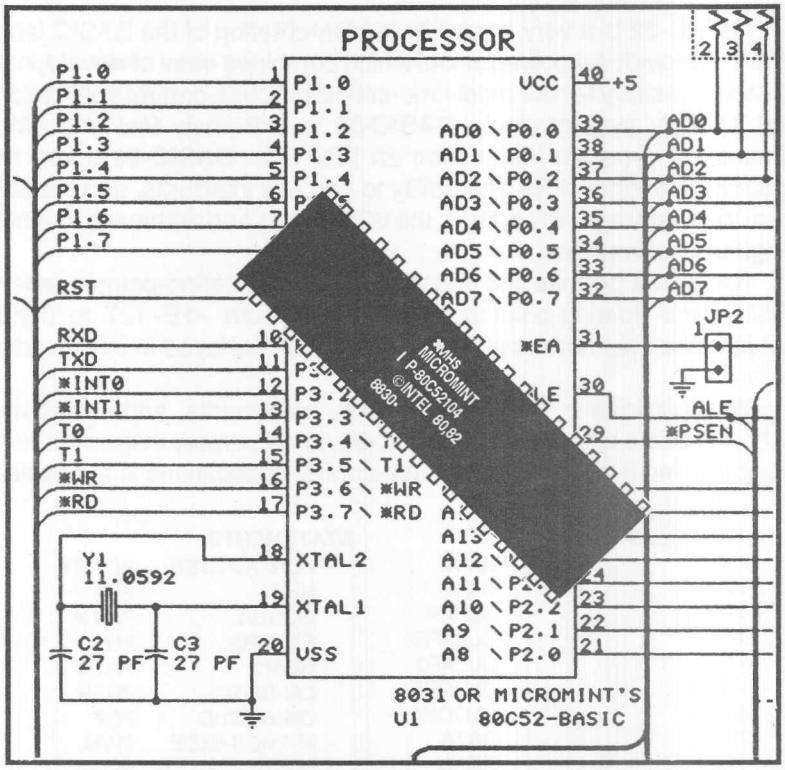
80C52-BASIC

The 80C52-BASIC chip is a custom-masked 80C52 microcontroller with a full-featured 8K-byte ROM-resident BASIC-52 interpreter. The 80C52-BASIC chip is specifically designed to address the needs of process control, measurement, and instrumentation applications. The internal BASIC-52 language allows users to write programs directly in BASIC. Assembly language routines are easily executed as CALL routines from BASIC.

The fully static design of the 80C52-BASIC chip allows the user to reduce system power by reducing the clock frequency from 12 MHz down to any value, even DC, without loss of data or internal registers (typical operating frequency for BASIC-52 is 11.0592 MHz). In addition, the 80C52 has two software modes for reduced activity: Idle Mode, where the CPU is frozen but the serial port, timers, and interrupt system continue to function; and Power-Down Mode, where the internal RAM is saved but all other functions are ceased.

A minimum amount of hardware is required to support the 80C52-BASIC chip. Small systems can be constructed with only an address latch, 1K byte of external memory, and the appropriate serial port drivers. With the addition of a transistor, a gate, and a few passive components, BASIC-52 can program EPROM/EEPROMs directly. Both standard and fast programming algorithms are supported.



Hardware

- Low-power fully CMOS static design
- Operates DC to 12 MHz
- Commercial or Industrial temperature range
- Jumper-selectable ROMless 80C31/80C32 operation
- Full-duplex serial port
- Three 16-bit timers
- 5-source 2-level interrupt structure
- On-chip oscillator and clock circuit
- 256 x 8-bit RAM

Software

- Full BASIC interpreter in ROM on a single chip
- BCD floating-point math
- Fast tokenized interpreter
- "Stand-alone" software development
- Interrupts can be handled by BASIC or Assembly Language
- Built-in real-time clock
- Generates all timing necessary to program EPROM and EEPROMs

MICROCONTROLLERS

The BASIC-52 Interpreter

BASIC-52 is a very complete implementation of the BASIC language contained in just 8K bytes of ROM. It provides a powerful tool which combines ease of development in a high-level language with the speed necessary for the most time-critical process-control applications. Due to the low system overhead and 11-MHz system clock, **BASIC-52** is extremely fast and efficient. It actually runs the "Sieve" benchmark program faster than an IBM PC. **BASIC-52** offers many unique features, including an accurate real-time clock, the ability to process interrupts, and the ability to treat EPROM memory as if it were mass storage. In addition, the I/O routines and arithmetic routines in BASIC are callable as assembly language subroutines.

BASIC-52 permits use of both integer and floating-point numbers. Integer numbers range from 0 to 65535, and floating-point numbers range from –1E–127 to 0.99999999E+127 with eight digits of significance. Numbers may be entered and displayed in integer, decimal, hexadecimal, or exponential format.

The following is a list of comands, statements, and operators supported by the **80C52-BASIC**. Although some are unique to **BASIC-52**, just a cursory inspection reveals that the full power of structured programming in BASIC for process-control applications is available.

COMMANDS

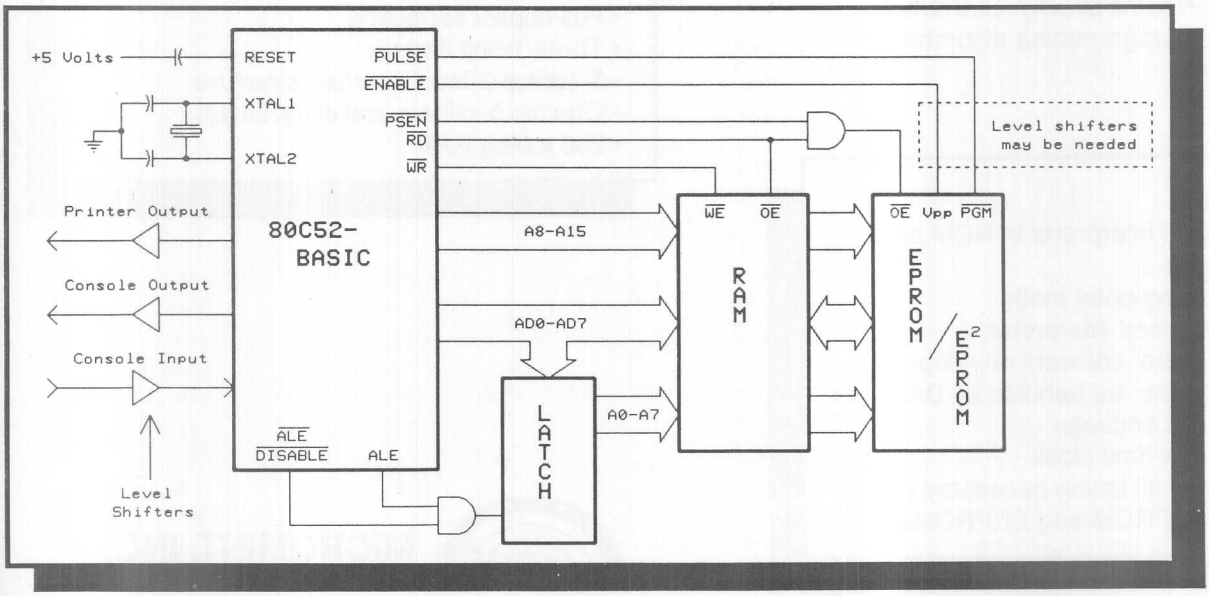
RUN
LIST
LIST#
NEW
NULL
RAM
ROM
XFER
PROG
PROG1
PROG2
FPROG
FPROG1
FPROG2

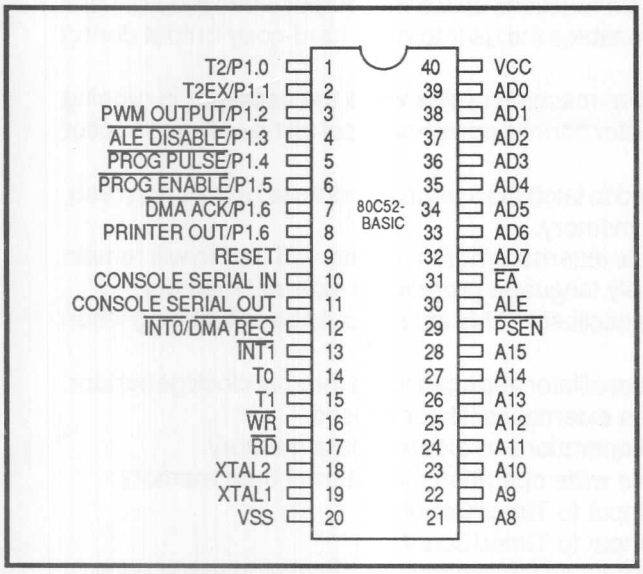
STATEMENTS

BAUD
CALL
CLEAR
CLEARS
CLEAR1
CLOCK0
CLOCK1
DATA
READ
RESTORE
DIM
DO-WHILE
DO-UNTIL
END
FOR-TO-STEP
NEXT
GOSUB
RETURN
GOTO
ON-GOTO
ON-GOSUB
IF-THEN-ELSE
INPUT
LET
ONERR
ONEXT1
ONTIME
PRINT
PRINT#
PH0.
PH0.#
PH1.
PH1.#
PUSH
POP
PWM
REM
RETI
STOP
STRING
UO0
UO1
UI0
UI1

OPERATORS

\+
\-
*
/
**
.AND.
.OR.
.XOR.
NOT
ABS()
INT()
SGN()
SQR()
RND
LOG()
EXP()
PI
SIN()
COS()
TAN()
ATN()
=
<
<=
>
>=
<>
ASC()
CHR()
CBY()
DBY()
XBY()
GET
IE
IP
PORT1
PCON
RCAP2
T2CON
TCON
TMOD
TIME
TIMER0
TIMER1
TIMER2
XTAL
MTOP





Operating Conditions*

Operating Temperature:
Commercial 0°C to 70°C
Industrial –40°C to 85°C

Operating voltage (Vcc): +5 volts ± 10%

Absolute Maximum Ratings*

Voltage on any pin with respect to ground (Vss): –0.5V to 7.0V

Power dissipation: 200 mW

Maximum Icc at 12 MHz: 24 mA

80C52-BASIC Pin Description*

Vss — Circuit ground potential

Vcc — Circuit supply voltage

AD0–AD7 — The multiplexed low-order address and data bus used during access to external memory. External pull-up resistors (10k ohm) are required on these pins if BASIC-52 EPROM/EEPROM programming feature is used.

A8–A15 — The high-order address bus used during access to external memory.

PORT 1 — Port 1 is a quasi-bidirectional 8-bit input/output port. It can be used as a standard parallel I/O port with the PORT1 command in BASIC-52, or the individual pins of Port 1 can have alternative functions as follows:

PORT 1.0 (T2) — Can be used as the trigger input to Timer/Counter #2. A logic 1 must be written to this bit in order for this function to operate.

PORT 1.1 (T2EX) — Can be used as the external input to Timer/Counter #2. A logic 1 must be written to this bit in order for this function to operate.

PORT 1.2 (PWM OUTPUT) — This pin is used as the Pulse Width Modulated (PWM) output port when the PWM statement is executed. The PWM statement can generate pulses of varying frequency and duty cycle.

PORT 1.3 (ALE DISABLE) — This pin is used to disable the ALE signal to the external latch when the EPROM/EEPROM programming feature is being used. In a system, this pin is logically ANDed with ALE.

PORT 1.4 (PROGRAMMING PULSE) — This pin provides the proper programming pulse when programming EPROM/EEPROMs.

PORT 1.5 (PROGRAMMING ENABLE) — This pin is used to enable the programming voltage (Vpp) when programming EPROMs and remains active low during programming. On EEPROMs that do not require any special programming voltage, this pin is not used.

PORT 1.6 (DMA ACKNOWLEDGE) — When the pseudo-DMA feature is implemented (as outlined in the BASIC-52 Programmer's Manual), this pin functions as an active-low DMA Acknowledge output.

PORT 1.7 (LINE PRINTER OUTPUT) — This pin functions as a serial output when the LIST# or PRINT# commands are used in BASIC. This enables the user to have hard-copy output during program operation or for program listings.

RESET — A logic 1 (>3.5V) on this pin for more than two machine cycles while the oscillator is running willl reset the device. An internal pull-down resistor permits power-on reset using only a capacitor connected between this pin and Vcc.

ALE — (Address Latch Enable) an output pin that is used to latch the low-order address byte during read, write, or program fetch operations to external memory.

PSEN — (Program Store Enable) a signal used to enable external program memory. This pin will remain a logic 1 unless the user is running an assembly language program in external memory.

XTAL1 — Input to the inverting amplifier that forms the oscillator. This input should be left floating when an external oscillator is used.

XTAL2 — Output of the inverting amplifier that forms the oscillator and input to the internal clock generator. Receives the external oscillator signal when an external oscillator is used.

$\overline{RD}$ — This pin is a control that is used to enable read operations to external data memory.

WR — This pin is a control signal that is used to enable write operations to external data memory.

T1 — This pin can be programmed to be an external input to Timer/Counter 1

$\overline{T0}$ — This pin can be programmed to be an external input to Timer/Counter 0

INT1 — This is the external interrupt 1 input pin. Interrupts on this pin may be handled in either BASIC-52 or assembly language.

INT0/DMA REQUEST — This is the external interrupt 0 input pin. It may optionally be programmed to function as a DMA request input pin or used by EEPROM devices during programming.

CONSOLE SERIAL OUTPUT — This is the serial input pin that receives data from the console device. Standard serial ASCII codes consisting of 8-bit data with no parity at standard data rates are assumed. After RESET in BASIC-52, if desired and if the first character received is a "space," then BASIC-52 will perform an auto-baudrate calculation and automatically set the console serial input to the incoming data rate.

$\overline{EA}$ — When EA is held high, the CPU functions as an 80C52 with a BASIC interpreter executing out of internal program memory (unless the program counter exceeds 0FFFh). When EA is held low, the CPU functions as a generic 80C52 and executes only out of external program memory.

*** Note:** For complete details of the 80C52-BASIC chip's electrical characteristics and timing diagrams, refer to the design specification sheet for the 80C52 microcontroller chip.